# 日期时间表

**1，创建函数，方便调用。**

**2，创造一个日期对象，获取当前的日期时间。为他设置变量。**

var date=new Date();

**3，通过命令调取日期对象时间。并设置变量储存。**

var year=date.getFullYear();

var month=date.getMonth()+1;

var day=date.getDate();

var hour=date.getHours();

var minute=date.getMinutes();

var second=date.getSeconds();

var week=date.getDay();//0—6

为了避免时间变为单位数后空间的改变，为他们添加一个零0

if( second<10){

second="0"+second

}

if( minute<10){

minute="0"+minute

}

if( hour<10){

hour="0"+hour

}

if( day<10){

day="0"+day

}

if( month<10){

month="0"+month

}

**4，将所需要的变量统一起来统一命名 为新的变量。**

var showTime=year+"年"+month+"月"+day+"日"+hour+":"+minute+":"+second;

**5，在文档中根据ID查找元素。并设置变量**

var o=document.getElementById("time");

**6将组合时间内容显示在该ID的尖括号内**o.innerHTML=showTime;

**7，在函数外设置函数的调用，与函数的刷新值。**

updateTime();//调用日期函数

setInterval(updateTime,1000);//根据固定的时间间隔执行某段代码函数

# 计算器

1. **在body中设置input和option，lable与button标签，并对input和option与button设置ID**

<input type="text" name="" id="op1">

<select style="width:50px; height:30px;" id="cal">

<option>+</option>

<option>-</option>

<option>\*</option>

<option>/</option>

</select>

<input type="text" name="" id="op2">

<lable>=</label>

<input type="text" name="" id="txt\_res">

<button id="btn">计算</button>

1. **在JS中，将button的ID赋值给变量**

var btn=document.getElementById('btn');//内容

1. **对button设置鼠标点击触发时触发函数。**

btn.onclick=function(){//

1. **函数中对前面三个ID的内容，进行赋值到变量，并创建为、未赋值变量。**

var value1=document.getElementById('op1').value;

var value2=document.getElementById('op2').value;

var cal=document.getElementById('cal').value;

var result;

1. **创建分支语句，设定当option取不同值时，输入数值的input间的运算方法。**

if(cal=="+"){

result=parseInt(value1)+parseInt(value2);//parseInt只是加值

}else if(cal=="-"){

result=value1-value2;

}else if(cal=="\*"){

result=value1\*value2;

}else if(cal=="/"){

result=value1/value2;

}

1. **将通过分支语句赋值过后的变量赋值给文档中的最后一个input的ID；**

document.getElementById('txt\_res').value=result;

**文档对象模型DOM（Document Object Model）定义访问和处理HTML文档的标准方法。DOM 将HTML文档呈现为带有元素、属性和文本的树结构（节点树）。**



1. 元素节点：上图中<html>、<body>、<p>等都是元素节点，即标签。

2. 文本节点:向用户展示的内容，如<li>...</li>中的JavaScript、DOM、CSS等文本。

3. 属性节点:元素属性，如<a>标签的链接属性href="http://www.imooc.com"。

**通过ID获取元素**

document.getElementById(“id”) （文档中的获取元素为ID）innerHTM指的是div的尖括号内内容

<p id="con">JavaScript</p>

<script type="text/javascript">

var mychar=document.getElementById('con'); ;

document.write("结果:"+mychar); //输出获取的P标签。

</script>

# 杨辉三角

设置行列

function demo10(n){

//控制行

for(var i=1;i<=n;i++){//3

for(var j=1;j<=i;j++){//2

document.write(cal(i,j)+"&nbsp;&nbsp;&nbsp;&nbsp;"); 引用函数并输出

}

document.write("<br>");

}

}

设置运算函数

//计算行为m,列号为n的位置的数字

function cal(m,n){

//每一行第一列

if(n==1){ 第一列为1

return 1;

}else if(m==n){ 最后一列为1

//每一行最后一列

return 1;

}else{ 若不满足则内调本函数，计算上一行的相同列与前一列的数值和。

return cal(m-1,n)+cal(m-1,n-1);//递归

} 递归算法为对括号内的函数继续调用

} 最终为计算一个数为将之前的所有数进行重新计算。

demo10(10);

# 确认一直弹出文字，取消停止弹出文字

<head>

<meta charset="UTF-8">

<title>Document</title>

<style type="text/css">

#box{margin:50px auto; }

button{ padding: 10px 20px; } 对标签设置简单样式

#box #show{width:200px;height:1200px;color:red;border: 1px solid green; }

</style>

</head>

<body>

<div id='box'>

<button id='start'>开始</button>

<button id='stop'>停止</button> 添加标签并设置ID

<div id='show'></div>

</div>

</body>

<script type="text/javascript">

var butStart=document.getElementById('start');

var butStop=document.getElementById('stop');

var showMe=document.getElementById('show'); 一一找到相对应的ID

butStart.onclick=function (){ 设置点击事件的匿名函数

t=setInterval( 设置定时器

function(){ 定时函数，输出内容与换行

showMe.innerHTML=showMe.innerHTML+'你好<br>';

},50); 触发函数的时间

}

butStop.onclick=function(){clearInterval(t);}; 点击事件时关闭定时器

</script>

# 三元操作符

<script type="text/javascript">

/\*var a=true;

if(a){

alert("真");

}else{

alert("假");

}\*/

var a=false;

//三元操作符 表达式?结果1:结果2

//若表达式结果为true，则执行结果1，否则执行结果2

a?alert("真"):alert("假");

</script>

# 增加，移除事件

<script type="text/javascript">

window.onload=function(){

var btn=document.getElementById("btn");

//添加事件

document.getElementById("btnAdd").onclick=function(){

// alert("添加事件");

btn.addEventListener("click",show,true);

};

//移除事件

document.getElementById("btnRemove").onclick=function(){

// alert("添加事件");

btn.removeEventListener("click",show,true);

};

function show(){

alert("我可以被点击了");

}

};

</script>

</head>

<body>

<button id="btn">按钮</button>

<button id="btnAdd">添加事件</button>

<button id="btnRemove">移除事件</button>

</body>

<style type="text/css">

#end{

display: none;

}

</style>

<script type="text/javascript">

var o\_show;

var o\_over;

window.onload=function(){

o\_show=document.getElementById("time");

o\_over=document.getElementById("end");

init();

};

function init(){

//倒计时 12:00:00

var endDate=new Date("2017/6/9 11:01:00");//结束事件

var nowDate=new Date();

var diffTime=(endDate.getTime()-nowDate.getTime())/1000;//秒

if(diffTime<0){

o\_show.style.display="none";

o\_over.style.display="block";

}

var hour=parseInt(diffTime/(60\*60));//小时

var minute=parseInt((diffTime/60)%60);//分

var second=parseInt(diffTime%60);//秒

if(hour<10){

hour="0"+hour;

}

if(second<10){

second="0"+second;

}

if(minute<10){

minute="0"+minute;

}

var o\_h=document.getElementById("\_hour");

var o\_m=document.getElementById("\_minute");

var o\_s=document.getElementById("\_second");

//将时间显示在页面中

o\_h.innerHTML=hour;

o\_m.innerHTML=minute;

o\_s.innerHTML=second;

setTimeout(init,1000);

}

</script>

</head>

<body>

<div id="time">还剩余：<label id="\_hour">1</label>小时<label id="\_minute">40</label>分<label id="\_second">40</label>秒</div>

<div id="end">秒杀结束</div>

</body>

# 封装JS库

function $(selector){//创造新对象函数，并输入值；

return new Base (selector);//返回新的对象

};

var Base=function(selector){//创造对象，判断输入的值

this.elements=[];//添加属性

if(selector!=undefined){//判断输入值是否是有值

if(typeof selector=="object"){//判断若输入值得类型为对象

this.elements[0]=selector;//

}

else{ //若输入值不是对象类型

var c=selector.charAt(0);//将查到的第一个字符串存储在变量C中

if(c=="#"){ //若C的内容为#

var s=selector.substring(1)//提取从第二个字符到最后之间的内容；

this.getId(s); //调用getId属性

}else if(c=="."){ //若C的内容为.

var s=selector.substring(1);//提取从第二个字符到最后之间的内容；

this.getClassName(s);//调用getClassName属性

}else{ //若以上都不满足

this.getTagName(selector); //调用getTagName属性

}

}

}

};

Base.prototype.getId=function(id){//定义属性

var o=document.getElementById(id);//将获取到的id储存在变量O中

this.elements.push(o); //将变量o添加到base的elements属性中

return this; //返回base对象

};

Base.prototype.getClassName=function(className){//定义属性

var o=document.getElementsByClassName(className);//将获取到的id储存在变量O中

for (var i=0;i<o.length;i++){//对变量o进行遍历

this.elements.push(o[i]);//将数组o的每一个添加到base的elements属性数组中

};

return this; //返回base对象

};

Base.prototype.getName=function(name){

var o=document.getElmentsByName(name);

for(var i=0;i<o.length;i++){

this.elements.push(o[i]);

};

return this;

};

Base.prototype.getTagName=function(tagName){

var o=document.getElementsByTagName(tagName);

for(var i=0;i<o.length;i++){

this.elements.push(o[i]);

};

return this;

};

Base.prototype.html=function(htmlValue){//定义base的HTML属性

if(arguments.length==0){ //判断输入值得长度可得知是否输入内容，为0则代表未输入值

return this.elements[0].innerHTML;//返回这个属性的第一个数组的尖括号之间的内容

}else{ //若是输入的有内容

for(var i=0;i<this.elements.length;i++){//遍历base的elements属性

this.elements[i].innerHTML=htmlValue;//对属性数组中的每一个对象的尖括号之间添加内容，为所输入的内容

};

}

return this; //返回对象base

};

Base.prototype.css=function(prototype,values){//定义base的css属性

if(values!=undefined){

for(var i=0;i<this.elements.length;i++){

this.elements[i].style[prototype]=values;

};

}else if(typeof prototype=="object"){

for(var i=0;i<this.elements.length;i++){

for(var j in prototype){

this.elements[i].style[j]=prototype[j]

}

}

}else{

var o=this.elements[0];

var res=o.style[prototype];

res=res==""?window.getComputedStyle(o,null)[prototype]:res;

return res;

}

return this;

};

Base.prototype.attr=function(prototy,value){

if(value!=undefined){

for(var i=0;i<this.elements.length;i++){

this.elements[i][prototy]=value;

}

}else{

return this.elements[0][prototy];

}return this;

};

Base.prototype.click=function(fn\_click){

for(var i=0;i<this.elements.length;i++){

this.elements[i].onclick=fn\_click;

};return this;

};

Base.prototype.hover=function(fn\_over,fn\_out){

for(var i=0;i<this.elements.length;i++){

this.elements[i].onmouseover=fn\_over;

this.elements[i].onmouseout=fn\_out;

};return this;

};

Base.prototype.show=function(){

for(var i=0;i<this.elements.length;i++){

this.elements[i].style.display="block";

};

};

Base.prototype.hide=function(){

for(var i=0;i<this.elements.length;i++){

this.elements[i].style.display="none";

};

};

# 注册页面

<script type="text/javascript">

function checkForm(){//返回运行三个检查函数

return checkuName()&&checkPass()&&checkPhoneNumber();//若三个函数全返回为true则执行跳转网页

};

function checkuName(){//检查用户名

var p=/^[a-zA-Z]\w{5,11}$/;//用户名开头为字母，并在6-12位之间

var uNma=document.getElementById('uNma'); //获取用户名id

var uNmaError=document.getElementById("uNmaError");//获取用户名报错文字ID

console.log(p.test(uNma.value));

console.log(!p.test(uNma.value));

if(!p.test(uNma.value)){//使用正则函数P检测括号内字符串，匹配返回true，不匹配返回false (此处不等！意为false==)

uNmaError.style.opacity=1;//用户名报错ID的透明度更改为0

uNma.focus();//元素的文字被选择

uNma.select();//元素的选择框被选中

return false;//返回false则停止运行

};

return true;//

}

function checkPass(){//检测密码的函数

var pass=document.getElementById("pass");

var repass=document.getElementById("repass");

var passError=document.getElementById("passError");

var repassError=document.getElementById("repassError");

var p2=/\w{6,12}$/;//正则函数字符类型的在6-12位之间

console.log(2);

if(!p2.test(pass.value)){//使用p2判断密码输入框的内容是否符合p2正则函数，匹配返回true，不匹配返回false (此处不等！意为false==)

passError.style.opacity=1;//密码错误提示ID的透明度为1

pass.select();

pass.focus();

return false;

}

if(pass.value!=repass.value){

repassError.style.opacity=1;

repass.select();

repass.focus();

return false;

}

return true;

}

function checkPhoneNumber(){

var phoneNumber=document.getElementById("phoneNumber");

var phoneNumberError=document.getElementById("phoneNumberError");

var p3=/[0-9]{11,}/;

console.log(3);

if(!p3.test(phoneNumber.value)){

phoneNumberError.style.opacity=1;

phoneNumber.select();

phoneNumber.focus();

return false;

}

return true;

};

</script>

<body><!--//提交表单时提交到网页solog.html；

浏览器使用 method 属性设置的方法将表单中的数据传送给服务器进行处理。共有两种方法：POST 方法和 GET 方法。

如果采用 POST 方法，浏览器将会按照下面两步来发送数据。首先，浏览器将与 action 属性中指定的表单处理服务器建立联系，一旦建立连接之后，浏览器就会按分段传输的方法将数据发送给服务器。

在服务器端，一旦 POST 样式的应用程序开始执行时，就应该从一个标志位置读取参数，而一旦读到参数，在应用程序能够使用这些表单值以前，必须对这些参数进行解码。用户特定的服务器会明确指定应用程序应该如何接受这些参数。

另一种情况是采用 GET 方法，这时浏览器会与表单处理服务器建立连接，然后直接在一个传输步骤中发送所有的表单数据：浏览器会将数据直接附在表单的 action URL 之后。这两者之间用问号进行分隔。

一般浏览器通过上述任何一种方法都可以传输表单信息，而有些服务器只接受其中一种方法提供的数据。可以在 <form> 标签的 method （方法）属性中指明表单处理服务器要用方法来处理数据，使 POST 还是 GET。

onsubmit事件在表单被提交时执行-->

<form action="solog.html" method="post" onsubmit="return checkForm()">

<h2>用户注册</h2>

<table>

<tr>

<td><label>用户名：</label></td>

<td><input id="uNma" type="text" name=""></td>

<td><span id="uNmaError">用户名以字母开头，6-12位</span></td>

</tr>

<tr>

<td><label>密码：</label></td>

<td><input id="pass" type="password" name=""></td>

<td><span id="passError">密码6-12位</span></td>

</tr>

<tr>

<td><label>确认密码：</label></td>

<td><input id="repass" type="password" name=""></td>

<td><span id="repassError">密码与上面的不一致</span></td>

</tr>

<tr>

<td><label>请输入手机号码：</label></td>

<td><input id="phoneNumber" type="text" name=""></td>

<td><span id="phoneNumberError">输入号码不正确</span></td>

</tr>

<tr>

<td colspan="1"><input id="radio" type="checkbox" name="" style="width: 20px"></td>

<td><a href="#">腾讯安全协议</a></td>

<td></td>

</tr>

<tr>

<td colspan="2"><input id="but" disabled="disabled" type="submit" style=" width: 80px;"></td>

<!--disabled属性 默认按钮为不可选用 -->

</tr>

</table>

</form>

</body>

<script type="text/javascript">

var radio=document.getElementById("radio");

var but=document.getElementById("but");

radio.onclick=function(){//单机按钮的点击事件

var che=this.checked?"":"disabled";//判断选择框的选项属性，若为true则返回空字符，若为false则返回disabled

but.disabled=che;//设置按钮的显示属性随着选择框的改变而改变

}

</script>

# 画布饼状图

<div><canvas id="cvs" width="500" height="500"></canvas></div>

</body>

<script type="text/javascript">

var o=[{

pro:0.3,

color:"red",

txt:"水果"

},{

pro:0.4,

color:"green",

txt:"面食"

},{

pro:0.1,

color:"blue",

txt:"喝水"

},{

pro:0.2,

color:"pink",

txt:"零食"

}];

var cvs=document.getElementById('cvs');//获取画布ID

var ctx=cvs.getContext("2d");//获取2d属性

var cx=250;//原点X

var cy=250;//原点Y

var r=100; //半径

var startAngle=0;//开始角度

for(var i=0;i<o.length;i++){//遍历整个数组

ctx.beginPath();//开始新路径

ctx.fillStyle=o[i].color;//设置填充颜色为某个数组中对象的属性

var endAngle=o[i].pro\*360;//结束角度为数组对象属性乘360度

ctx.moveTo(cx,cy);//设置画笔的起始位置为原点

ctx.arc(cx,cy,r,startAngle\*Math.PI/180,(startAngle+endAngle)\*Math.PI/180);

//画圆，以原点为圆心，半径为r，开始角度为起始角度\*PI（半圆）除以180，得到的是以PI为单位的弧度，终止角度为起始角度加结束角度\*PI除以180，得到的是以PI为单位的弧度。

ctx.fill();//对所画的图形进行填充

var engle=startAngle+endAngle/2;//创建变量存储开始角度加上结束角度除以二刚好得到中线角度

if(engle>90&&engle<270){//如果度数大于90度小于270度则文字排列方式为右对齐

ctx.textAlign="right";

}else{

ctx.textAlign="left";//如果小于90度大于270度则文字排列方式为左对齐

}

var txtX=Math.cos(engle\*Math.PI/180)\*(r+10)+cx;//文字的X轴坐标为，原点坐标加上中线角度的COS值乘斜边得到的是x坐标；cos的角度90-270度为负值，270-90度之间为正值

var txtY=Math.sin(engle\*Math.PI/180)\*(r+10)+cy;//文字的Y轴坐标为，原点坐标加上中线角度的sin值乘斜边得到的是y坐标；sin的角度0-180度为负值，180-0度之间为正值

console.log(Math.cos(engle\*Math.PI/180),Math.sin(engle\*Math.PI/180));

ctx.fillText(o[i].txt,txtX,txtY);//填充文字，内容为对象属性，坐标为获得的坐标

startAngle=startAngle+endAngle;//初始化开始坐标

};

</script>